Question 0

Immagine che contiene testo

Descrizione generata automaticamente

The most difficult problem for a decision tree algorithm to learn was problem 2: indeed this tree classify the data based on less specific rules. Indeed, there are a lot more combinations that can happen for the “true” row of the second set compared to other two. It means you’re going to have to check every single feature and there will be less opportunity for pruning.

Question 1

Immagine che contiene testo

Descrizione generata automaticamente

🡪Entropy monkTraining 1= 1 🡪 this is the worst scenario where P(true) = P(false)

🡪Entropy monkTraining 2= 0.957 🡪 also very high, a not uniform distribution

🡪 Entropy monkTraining 3= 0.99 🡪 also bad, not uniform distribution

One can show that the entropy will take on a value near zero if the *p*s are all near zero or near one. Therefore, the entropy will take on a small value if the mth node is pure. When building a classification tree, either the Gini index or the entropy are typically used to evaluate the quality of a particular split, since these two approaches are more sensitive to node purity than is the classification error rate

Question 2

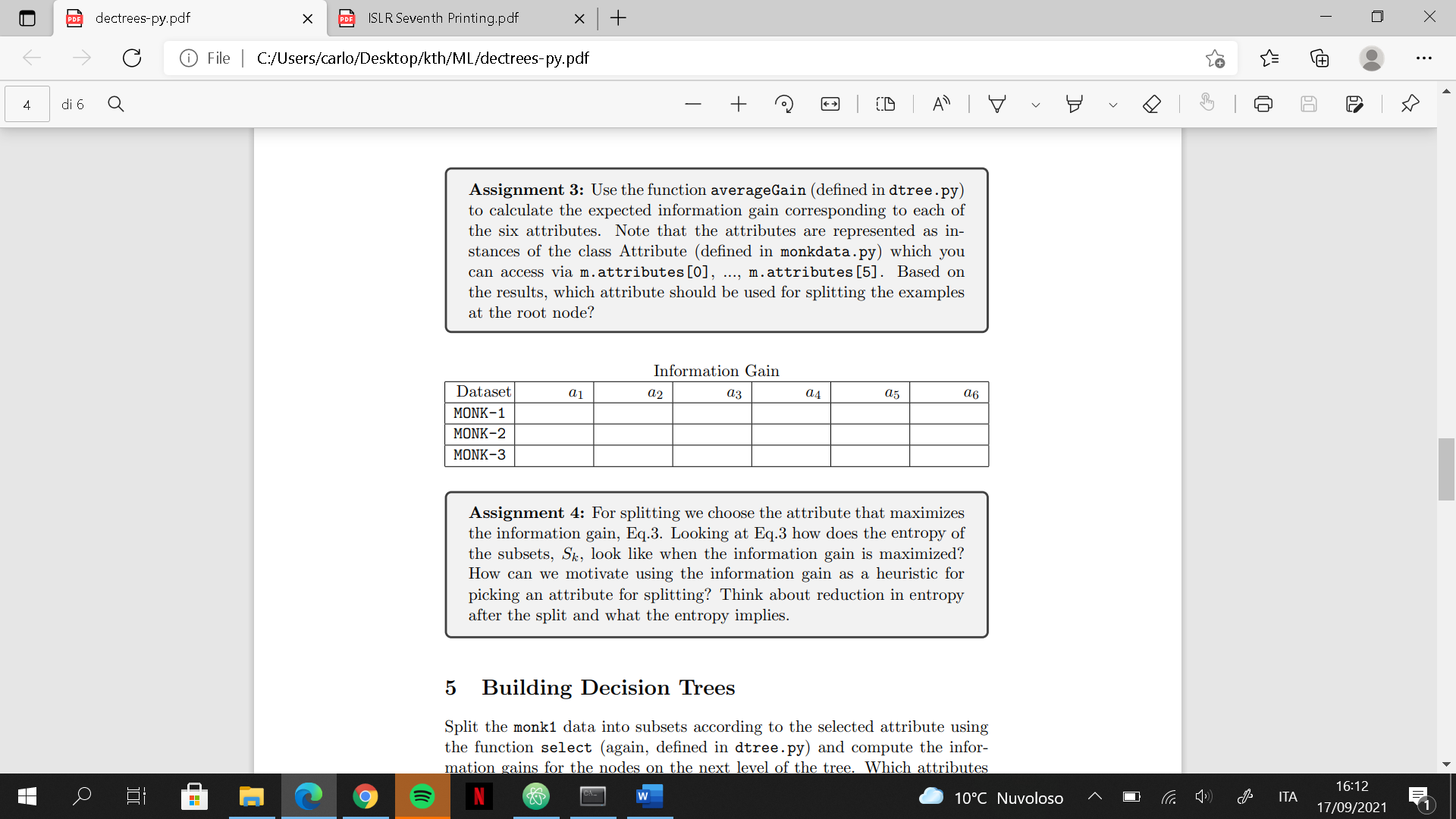
Immagine che contiene testo

Descrizione generata automaticamente

Uniform distributions have higher entropy compered with non-uniform distribution, indeed if we look at the formula of entropy, if the probability of being class A is equal to the proability of being class B 🡪 then entropy is going to be higher compared to if they had differ probabilities. The more different the probabilities are, the lower the entropy (ex p1= 0.9, p2=0.1).

An example could be a deck of cards where 20 of them are black and the other 20 are red. In distribution the entropy will be high because the two probabilities are the same. Instead if we have a deck of card with 5 reds and 35 blacks then the entropy will be much lower.

Question 3



For Monk1 🡪 a5 shows the highest gain so it will be the attribute chosen to make the first split

For Monk2 🡪 a5 as well

For Monk3 🡪 a2 as well

The information gain measures the expected reduction in impurity caused by partitioning the examples according to an attribute. It thereby indicates the effectiveness of an attribute in classifying the training data

Question 4

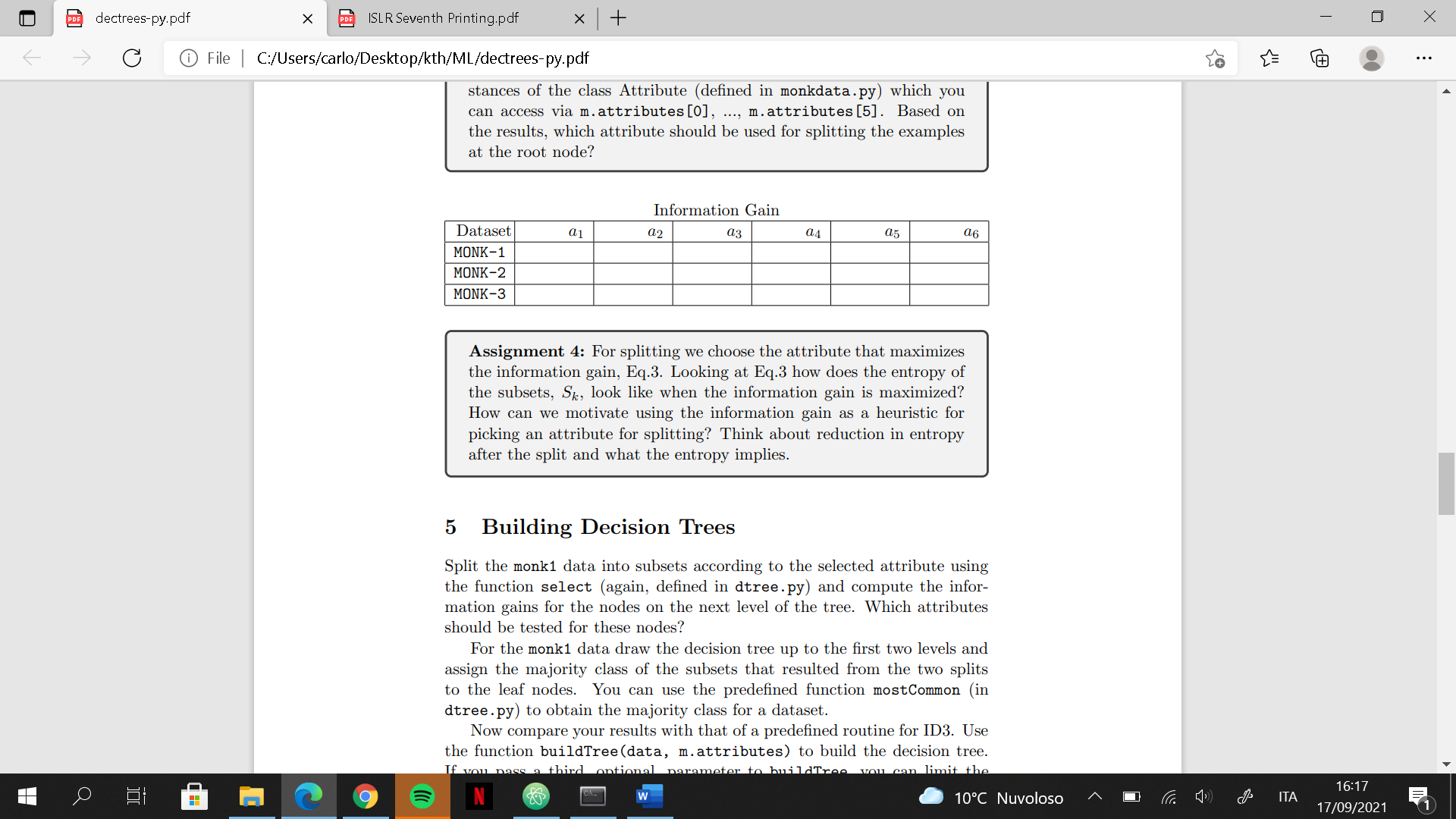


Immagine che contiene testo

Descrizione generata automaticamente

Since Gain = Total entropy – entropy for an attribute, it means that the attributes mentioned above have the lowest entropy, making them carry more information (Lower entropy means higher gain). Then to maximize the information gain Sk has to be minimized.

As a heuristic for piking an attribute for splitting we can use f(n)=maximized information gain because 🡪The attribute with the highest information gain is the attribute that for a certain split maximize the reduction of the impurity of two children nodes.

Building decision trees

Since attribute 4 was used to divide the initial node, 4 children nodes were created because a4 has 4 values. Node 1 is certain (a4 = 1). The attributes with the highest gains in nodes 2, 3 and 4 are attributes 4, 6 and 1, respectively. Since attributes 4, 6 and 1 can take a possible 3, 2 and 3 values, a total of 3 + 2 + 3 = 8 nodes (second layer) are created with values : FFFFFFFT. This is the same as when compared to the drawn tree with max depth = 2.

Tree picture

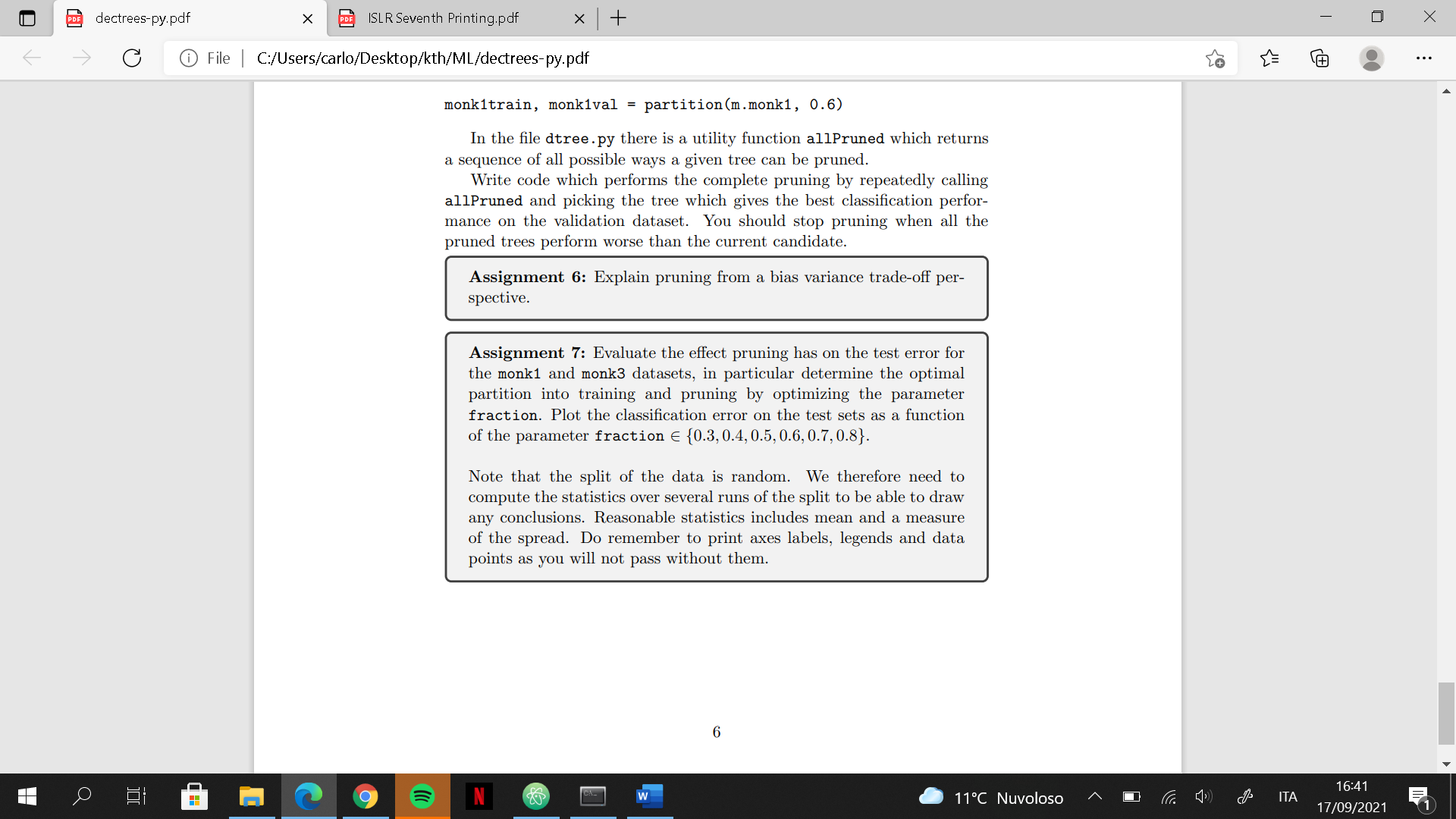
Question 5

Immagine che contiene testo

Descrizione generata automaticamente

Our assumptions on the datasets were correct, indeed the tree trained on the monk-2 dataset resulted to be the one with lowest accuracy. The accuracy obtained on the train datasets in each case is equal to 1: this is obvious since we trained the trees on the training data. On the other hand, the accuracy we obtain in the test data for Monk1-2-3 are respectively: 0.83, 0.69, 0.94 meaning that for the monk1Test data we can classify 83% of the observations correctly.

Question 6

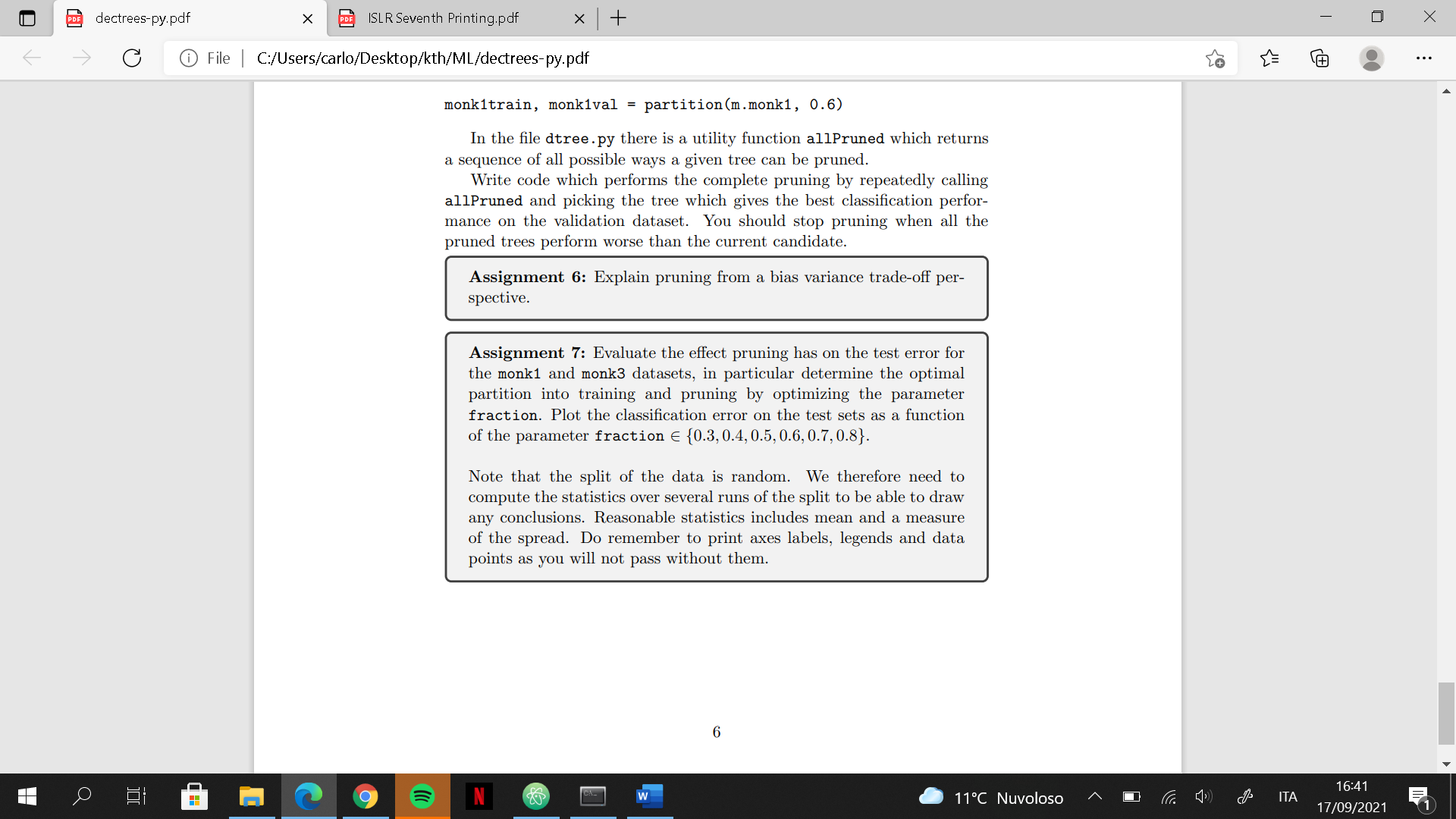


A good strategy to build a decision tree with high performances is to first grow a very large tree and then prune it back by removing some nodes to obtain a subtree. A node is removed if the resulting pruned tree performs at least as well as the original tree over a separate validation dataset. If a tree is too large and too complex it would overfit the data and will have bad performances on a test-data. By pruning a tree we want to get a tree that is smaller, meaning that it’s easier to interpret, and that performs better on a given test-data. In other words, we want a tree that has a higher variance (meaning that can perform well on new datasets) at cost of a little bias.

Diagram

Description automatically generated

Question 7



For the graphs we can see that the two fraction values that offer best performance (highest test-score and low standard deviation) are 0.5, 0.6, and 0.7. Having a high mean test-score means that they are performing well on the test-set, while having a low standard (in this cases) deviation means that for each iteration they keep performing well.